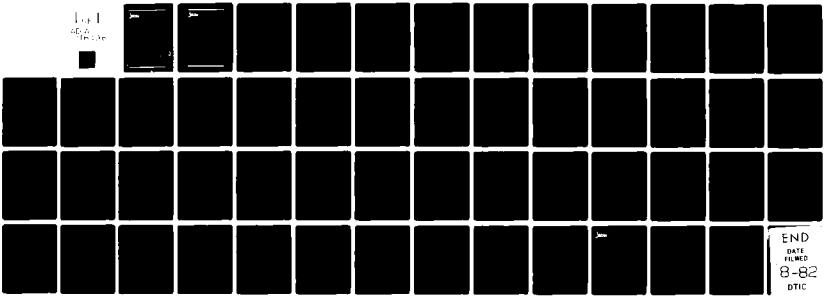
AD-A116 496 BDM CORP ALBUQUERQUE NM F/G 9/2
ANALYSIS OF SOFTWARE MAINTAINABILITY EVALUATION PROCESS.(U)
DEC 78 F29601-77-C-0082
UNCLASSIFIED BDM/TAC-78-698-TR NL



END
DATE FILMED
8-82
DTIC

(2)

THE BDM CORPORATION

THE BDM CORPORATION

1801 RANDOLPH ROAD, S.E. · ALBUQUERQUE, NEW MEXICO 87106 (505) 848-5000 · TWX 910-989-0619

ANALYSIS OF SOFTWARE MAINTAINABILITY EVALUATION PROCESS

6 December 1978

BDM/TAC-78-698-TR

FOREWORD

This report, BDM/TAC-78-698-TR, is submitted by The BDM Corporation, 2600 Yale Blvd, S.E., Albuquerque, New Mexico 87106 to the Air Force Test and Evaluation Center, Kirtland Air Force Base, New Mexico in response to reporting requirements of contract F29601-77-C-0082.

TABLE OF CONTENTS

ANALYSIS OF SOFTWARE MAINTAINABILITY EVALUATION PROCESS

A. INTRODUCTION

1. Scope

Paragraph 6.3.3 of Technical Directive Number 120 to Contract F29601-77-C-0082 requires the report of findings from tasks outlined in paragraph 4.2.1. These tasks are to determine:

(1) 4.2.1.1. If significant differences in results exist which are associated with types of application area for which a computer program is written on the relative size and complexity of the subject area but are independent of software maintainability considerations.

(2) 4.2.1.2. If the experience level, type of experience, functional knowledge or lack of functional knowledge of the program/module being evaluated has a bearing on the results independent of maintainability considerations,

(3) 4.2.1.3. The use and value of the comments sections of the questionnaires, and

(4) 4.2.1.4. Questions which are apparently being interpreted differently by different evaluators.

In addition to the above tasks, research of the validity of the maintainability model was also performed. This research included:

(1) Validation of the model parameters using factors analysis

(2) Validation of the model parameters using a survey of the computer science professionals most widely referenced in current computer science publications.

This report presents the analysis techniques which were utilized and the results obtained.

2. References

The following list of references includes all project-related deliverables as well as a typical reference for understanding the statistical techniques employed to analyze the software maintainability evaluation process.

(1) Ragland, F. and D. Peercy. Task Implementation Plan, BDM/TAC-78-010-TR, 6 January 1978.

This technical report presents the plans for implementing the project tasks. These include analyzing the current software maintainability methodology, making recommendations for changes to the methodology, and implementing a computer program to support the analysis of the evaluation data.

(2) Ragland, F. and D. Peercy. Interim Report, BDM/TAC-78-315-TR, 13 June 1978.

This technical report summarizes the analysis of the current software maintainability methodology and data from several previous software evaluations. Recommendations for changes in the methodology were also included.

(3) Peercy, D. Revised Test Plan, BDM/TAC-78-729-TR, 7 November 1978.

This technical report consisted of the software maintainability evaluation test plan to be included as part of the overall AFTEC software evaluation test plan. This test plan summarized the software maintainability evaluation methodology as revised from the recommendations contained in the Interim Report (see above).

(4) Peercy, D. Software Maintainability, Evaluator Guidelines Handbook, BDM/TAC-78-687-TR, 6 December 1978.

This technical report serves as the handbook for the Software Evaluation Team. This handbook contains a description of the general maintainability evaluation methodology and the specific evaluation procedure. Questionnaires for the documentation and source listing evaluations and various response suggestions, clarifications, and examples are also included.

(5) Peercy, D. and T. Paschich. Software Maintainability Analysis, Program Users Manual, BDM/TAC-78-697-TR, 6 December 1978.

This user's manual includes information to help the user of the Software Maintainability Analysis Program (SMAP) understand what the inputs and outputs (reports, diagnostics) of SMAP are. In addition, some general information concerning SMAP's flexibility is included.

(6) Peercy, D. and T. Paschich, Software Maintainability Analysis Program Maintenance Manual, BDM/TAC-78-696-TR, 7 December 1978.

This maintenance manual contains detailed design information which would be helpful to maintenance personnel in correcting errors or making modifications to the Software Maintainability Analysis Program (SMAP). The detailed information includes a description of the SMAP global data base and of each SMAP component and member module.

(7) Kerlinger, F. Foundations of Behavioral Research, 2nd Edition, Holt, Rinehart, and Winston, Inc., 1973.

(8) University of California, BMDP-77, University of California Press, 1977.

B. ANALYSIS TECHNIQUES

1. Data Screening

We desire an indication of the disagreement among raters in either of two situations. In the first, one individual, an outlier, is found to differ significantly from all others. If the distance between the outlier and the remaining homogeneous scores is great, the outlier will have a disproportionate affect on normal-theory measures of the distribution, such as the mean. The outlier alerts us to improper sampling or misunderstanding among the raters. Investigation can then lead to correction or proper interpretation of the results.

We also wish to know when the observations differ significantly overall, even though there may be no apparent outliers. In this situation, the population of scores is heterogeneous. Less confidence can be placed in the results because of this disagreement.

a. Outlier Detection

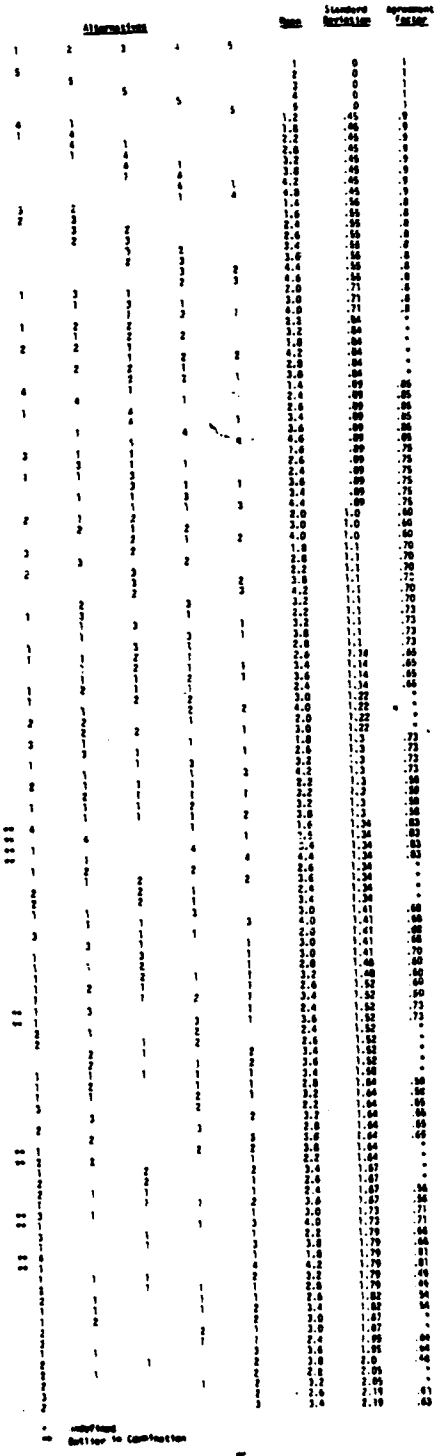
Table 1 lists all possible scoring combinations where five evaluators use a scale with five alternatives. The indicated combinations are considered to have unique observations far enough distance from the homogeneous group to be considered outliers.

Table 1 also includes the standard deviation and AFTEC agreement factor scores for each combination. Notice that the standard deviation does not provide an acceptably consistent measure for outliers or agreement. The agreement factor is not acceptable for determining outliers (e.g., the combination of four scores of 1 and one score of 4 has a relatively high agreement factor of .83, although it includes an obvious outlier).

The AFTEC agreement factor is calculated as:

$$A = \frac{1}{N} \sum_{i=0}^{NS-1} F_i/2^i \qquad (1)$$

TABLE 1. ALL SCORING COMBINATIONS OF FIVE RATERS FOR FIVE ALTERNATIVES



where:

A is the agreement factor

NS is the number of unit steps +1 in the scoring scale

F_i is the number of responses that are i steps from the mode

N is the number responses

A value of .7 appears to provide the best discrimination between acceptable and unacceptable agreement. The criteria for acceptability is a function of the analysis to be performed on the data. Analysis techniques differ with respect to their robustness in accounting for disagreement when providing results. Methods to be utilized in the software analysis program are highly robust with respect to error variance, which leads to the conclusion that an agreement factor of .7 or more is adequate.

If the question responses do not provide a unique statistical mode the agreement factor may be ambiguous. Table 1 lists combinations with ambiguous agreement factor scores. Combinations where the mode is undefined will be listed for reference as unacceptable. If the score distribution is bimodal and the two modes are adjacent, there is adequate agreement.

If adequate agreement does not exist, the outcome of the analysis must be questioned. Although analysis techniques to be utilized, such as analysis of variance, are robust with respect to error variance, excessive disagreement invalidates results. The analyst must be warned of this condition before drawing conclusions from the analysis results.

2. Reliability

When an attribute is measured, whether physical or physchological, the measurement contains chance error. Two sets of measurements of the same features will never exactly duplicate each other if the unit of measurement is fine enough in relation to the accuracy of the measurements. Unreliability means that repeated sets of measurements never exactly duplicate each other. At the same time, however, repeated measure-

ment of the same attribute will show some consistency. The tendency toward consistency from one set of measurements to another is called reliability.

When test results are interpreted, it is desirable to know how much the obtained score is likely to vary from a true measure of the attribute. Unreliability places a question mark after the score and causes judgment to be tentative. The lower the reliability, the more tentative any decision must be. In the extreme case, as reliability approaches zero, the score does not provide a basis for judgment.

The variation in a set of scores arises, in part, because of systematic differences between objects in the attribute being measured. This is referred to as "true" variance. In part, the variance also arises from unpredictable inaccuracies as the separate objects are measured. This is referred to as "error" variance. There are a number of statistics which have been developed to describe variability. The most useful of these for identifying the two forms of variability described above is the variance. An advantage of the variance is that it can be broken down into separate parts when the parts combine additively to give a total.

Designate the variance of the true scores of a group of evaluators as σ_T^2 and the variance of errors of measurement as σ_e^2. We assume that error is random and does not covary with the magnitude of the true score. If so, then

$$\sigma_x^2 = \sigma_T^2 + \sigma_e^2 \qquad (2)$$

That is, the variance of the obtained scores (σ_x^2) equals the variance of the true score (σ_T^2) plus the errors of measurement (σ_e^2).

Reliability can be defined through error: the greater the error, the lower the reliability. Alternately, the lower the error, the greater the reliability. Since we can measure total variance, if we estimate the error variance of a measure, we can also estimate reliability. Kerlinger (reference 7) observes that this brings us to two equivalent definitions of reliability:

(1) Reliability is the proportion of true variance to the total variance of the obtained scores.

(2) Reliability is the proportion of error variance to the total obtained score variance subtracted from 1.00, the index 1.00 indicating perfect reliability.

The definitions can be expressed as:

$$R = \frac{\sigma_T^2}{\sigma_x} \text{ and} \qquad (3)$$

$$R = 1 - \frac{\sigma_e^2}{\sigma_e^2} \qquad (4)$$

where R is the reliability coefficient. Since total error is observable, by obtaining error variance we can calculate reliability.

The statistical method for identifying error variance is Analysis of Variance (ANOVA). ANOVA allows the analyst to isolate the sources of variance within total variance. In the evaluation of module questionnaires, for example, the sources of variance are differences between the evaluators due to their differing backgrounds and expectations, differences in the characteristics of the modules, and unattributable differences due to error.

We wish to subtract the proportion of error variance to observed variance from 1.00 in order to arrive at reliability. Observed variance includes both variance due to differences between modules and variance due to differences between evaluators. If evaluator differences are removed, observed variance includes only differences between modules. Since we wish to determine a measure of reliability which is independent of evaluator differences, it is desirable to find the proportion of error variance to observed variance after removing evaluator effects.

Two-way analysis of variance allows a determination of all three variance sources. Mean-squares for raters, modules, and error are

determined as measures of variance. Reliability is then calculated as 1.00 minus the proportion of mean-square error to mean-square modules.

If the reliability coefficient R is squared (R^2), it becomes a coefficient of determination. It gives us the proportion of the variance shared by the "true" score and the observed score. R^2 is interpreted as the proportion of observed variance which can be attributed to a true measurement. The expressed $1 - R^2$ provides the proportion of total variance which can be attributed to error.

3. Regression

We desire to provide a stable method of evaluation which provides consistently accurate results across all evaluations. We must contend, however, with certain influences which run counter to this purpose.

(1) Criteria definitions, although carefully developed, may not be understood consistently in the same way by all evaluators. If not, something other than what was intended will be evaluated.

(2) Evaluation groups differ in background and ability. Where groups differ significantly, significant differences in evaluation results can occur. Such differences in outcome can lead to software rating acceptably by one group but not by another.

(3) Test environment and methods of performing the test will have strong influences on test outcome. It is desirable, therefore, to hold these influences constant so that all software is measured under the influence of the same external factors. Differences between tests in test methods or procedures, evaluator workload, or ease of assessment can, for example, cause unwanted variance (error) in the outcome.

(4) Although the maintainability characteristics being evaluated are comprehensive, they are not inclusive. As a consequence, desirable features present in a set of software may not benefit its evaluation by providing offset against those features which were identified in the test and on which it scored low. This interaction between test method and software tested will have differential effects on the outcome.

We wish to know when these influences have affected the outcome. One potential measure is provided by the statistical technique of multiple regression. Overall maintainability scores and scores for each criteria are filed by program. Over the course of several evaluations, a file develops which eventually allows the formulation of a mathematical model describing the best linear combination of the criteria scores which make up maintainability.

If there is a good fit between the model and the data, we have a means of predicting the maintainability score solely on the basis of the general question designed to evaluate each criteria. Obviously, one question per criteria will not provide the stability or accuracy necessary to rate the software, but it will provide a measure of whether the ratings were general as we would expect, given no strong influences from the error sources noted above.

A hypotheses test is performed to determine if there is significant difference between the actual and predicted maintainability scores. If the regression model fit to the data is good, as indicated by a high coefficient of determination value, significant differences between actual and predicted maintainability scores indicate that influences external to the software are not similar for past and present evaluators. The unwanted influence of one or more of the error sources discussed earlier has affected the results.

Although the use of a regression model will not identify which error source is causing difficulty, it alerts us to temper our conclusions when the need for such caution would otherwise be undetected.

4. Survey

As part of the analysis of the software maintainability evaluation process, The BDM Corporation conducted a survey of software professionals with background and interests in software quality assessment. This survey was sent to each of 200 software professionals and consisted of a one page cover letter explaining the request, a one page set of definitions of the AFTEC software maintainability hierarchy, and a

postcard with space for the respondent to reply with a rating for each software maintainability test factor's relative importance to software maintainability. Space was also provided for a short comment.

The objectives of the survey were to determine whether a set of software professionals with reasonably related interests could possibly agree on some universal set of test factor weights, and to solicit any comments concerning the general structure of AFTEC's software maintainability hierarchy. A sample of this survey is contained in Appendix B. The results of the survey are included in section C.

C. ANALYSIS RESULTS - IDENTIFYING REQUIRED CHANGES

1. Reliability

Reliability has been defined in terms of how well the software evaluators focus on the same characteristics. The test designer's ability to accomplish this depends both on design of the questionnaire and control of the test process. Thorough test planning, pretest instruction, guidelines for use during test, and posttest review for omission or mistake substantially improve rater understanding and reduce error. Accurate knowledge of the test designers intent leads to an improved agreement in rater focus.

Reliability is improved in the well executed test process because evaluators have an understanding of what is desired by the test designer and will agree in their interpretation of requirements. Reliability is improved in the well-executed test because evaluators understand the test designer's intent when questions are ambiguous. This understanding leads them to a more uniform interpretation of what is required.

Differences in evaluator focus can also be reduced by designing unambiguous questions. Emphasis on clarity must not stop here, however. Questions will be misunderstood or will lack clarity to some evaluators, no matter how careful the designer has been. It is critical, therefore, that requirements for both proper test execution and questionnaire design are satisfied to minimize wear.

Reliability changes due to improvement in test process and questionnaires are indicated in figure 1. Reliability scores for each question in AFTEC form 246 were averaged across system evaluations which were prior to implementing process changes (E-3A and F-16), system evaluations affected by process changes only (B-52CPT), and system evaluations affected by both process changes and redesigned questionnaires (F-16FMX). Cumulative percentages are indicated at various reliability levels.

The data from evaluations prior to process or questionnaire changes are indicated by the dotted line. For example, 74% of the questions had reliabilities lower than the 70-79 range prior to process changes. Reliabilities after process changes are indicated by dashes. Improvement has occurred. Only 63% of the reliabilities were below the 70-79 range after incorporating process changes. Reliabilities influenced by both process changes and the redesigned questionnaire are indicated by intermittent dots and dashes. Only 52% of the question reliabilities were below the 70-79 range after implementing both process and questionnaire changes. Appendix A presents supporting data.

Figure 1 indicates that to bring reliability up to the 60-69 percent range, redesign of the questionnaires did not show a significant difference over effects of the process changes. However, to get a higher proportion of reliability scores above the 60-69 range, the redesigned questionnaire was clearly required.

2. Types of Application Areas and Biodemographic Influences

Regression has been previously discussed as a method for determining effects of rater background on software assessment. This is accomplished by obtaining types of application areas and rater biodemographic characteristics and regressing them on an overall factor score for the software obtained from the raters. Hypothesis tests on the regression slopes will then indicate whether application areas or rater differences on a particular background variable lead to differences in scores.

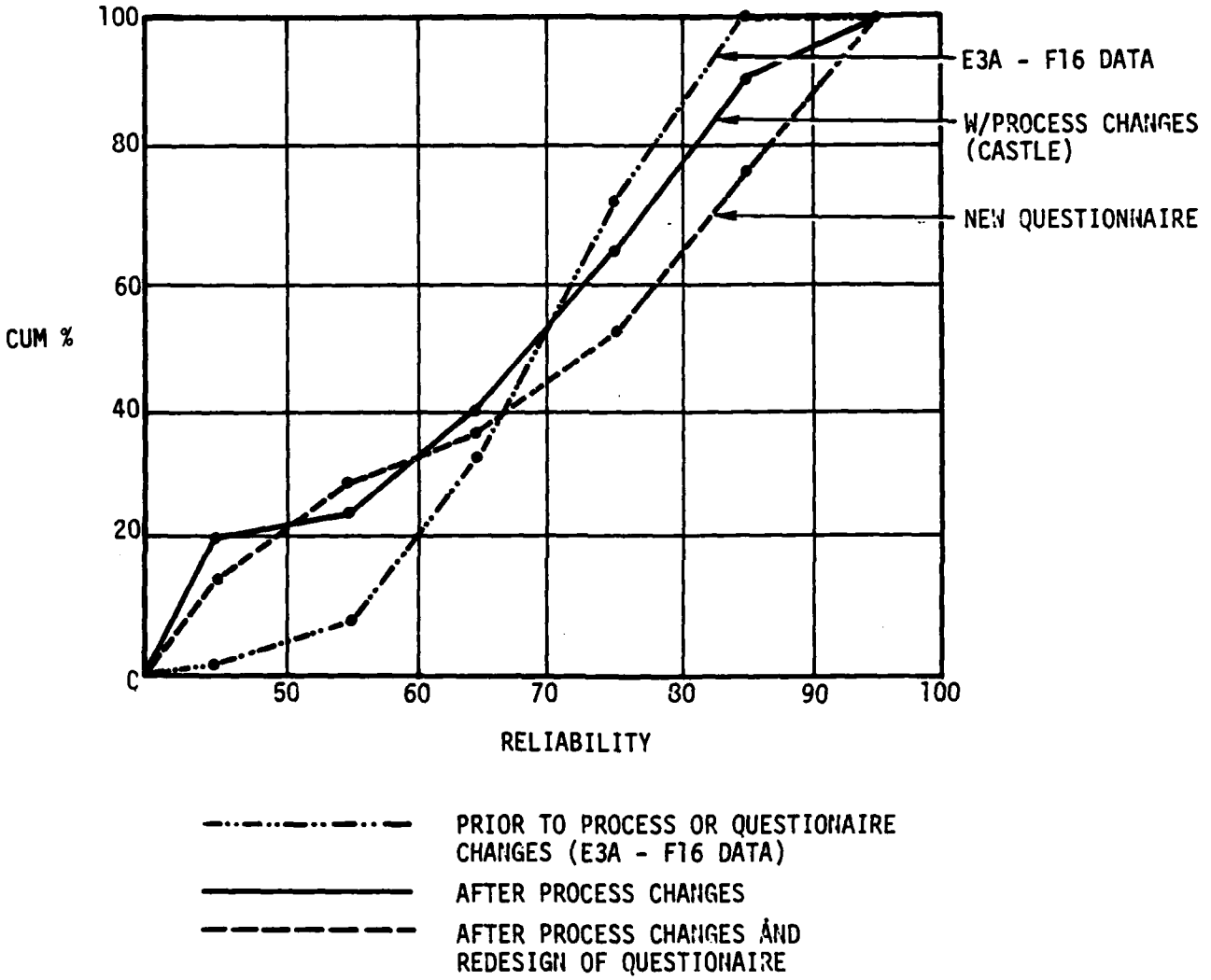


Figure 1. Comparative Questionnaire Reliabilities

Unfortunately, software tests did not occur at the rate originally planned and the extent of data required to obtain relationships is not presently available. As additional data become available, methods described above can be used to assess evaluator background affects.

3. Model Validity

A method was required to evaluate validity of the initial maintainability model. Factor analysis provides a means of evaluating relationships in the data. Questions can be identified by the factors in the theoretical model which they assess. If the model is accurate, questions addressing the same factor should result in data that are related. If the data do not demonstrate the expected relationships, other factors are influencing the outcomes and the model has failed to be validated.

Factor analysis was performed on data from the AFTEC form Q244 and Q245 questionnaires to assess the design and documentation elements of the model. The factor titled design structure was consistently confirmed with confirmation of design clarity occurring at a significantly lesser rate. *Other factors of the model were not confirmed.*

4. Evaluator Sample Size

We desire a sample size which insures that the imaginary population of all possible software raters is accurately represented by a randomly selected sample. We wish to guard against two possible errors. We do not wish to say the software is below a criteria when a much larger sample would find that the criteria was not (Type I error), and we do not wish to say the software meets standard when a much larger sample would find that it does not (Type II error). Consequently, we establish probabilities we are willing to accept for each of the two possible error types. The probability of a Type I error that we are willing to accept is termed alpha; a Type II error is termed beta. With alpha and beta defined, sample size n is given by

$$n = \left(\frac{Z_\alpha \sigma + Z_\beta \sigma}{\phi} \right)^2 \qquad (5)$$

where

Z_α = normal deviate at α

Z_β = normal deviate at β

σ = standard deviation of the population of scores

ϕ = not-to-exceed distance of the rater sample results from the hypothetical rater population results

Values of Z_α and Z_β are values found in most statistical texts for the normal deviate at areas under the normal curve specified by α and β. An α of .05, for example, results in a Z_α of 1.645.

The value of σ represents variance in the population of all possible software evaluation scores where comparable procedures were used. Since all possible software evaluations are not completed, a representative sample of variance in available scores is used to calculate the sample standard deviation, which is substituted for σ. The sample standard deviation is given by

$$s = \frac{\sum_{i=1}^{n} (X_i - \bar{X})^2}{n-1} \qquad (6)$$

where

s = sample standard deviation

n = sample size

X_i = evaluation scores in the sample

$\bar{X}$ = the mean of the samples scores

Since we realize that the method will not provide a sample size that will exactly satisfy all our desires for accuracy, we wish to specify how close the outcome must be to an outcome from a hypothetical survey of all possible raters. This value is specified for ϕ. If we wish to be no more than 0.5 distance from the outcome of a total survey of all possible raters, ϕ would be specified as 0.5.

The method described above can be used to evaluate required sample size at any time during future evaluations. The two objectives in determining the sample for calculating s are that it be representative of methods used in present testing and that as large a sample of prior test results as possible be included in its calculation.

Table 2 provides rater sample sizes calculated at $\alpha = .05$ and at various values of β. The F-16MX scores were used to calculate s.

Sample standard deviations for all questions of the source listing questionnaire were averaged across the four modules evaluated, resulting in an s of .733. Substituting into equation 5 for $\alpha = .05$, $\beta = .10$, and $\phi = .5$, we have

$$n = \left[\frac{(1.645)(.733)+(1.28)(.733)}{.5}\right]^2$$

$$= 18.39 \cong 19$$

The variance in the F-16FMX data was inflated by several influences which may be improved in future evaluations. When raters agree on what is being observed, fewer raters are required. The F-16FMX evaluation was affected by:

(1) The unavailability of complete evaluation guidelines.

(2) A minimal time for rater indoctrination prior to the evaluation.

(3) The presence of a strong individualist among the four available raters. This rater was a consistent contributor to outliers identified in the data.

(4) The requirement in the revised questionnaire to evaluate instrumentation. Four of 12 questions assessing instrumentation indicated unacceptable distributions. Additional instruction on what to evaluate is necessary to obtain agreement on what is observed.

Ten of 89 source listing questions had unacceptable distributions. Resolution of the problems noted above can be expected to

TABLE 2. SOFTWARE EVALUATION RATER SAMPLE SIZE CALCULATION - F-16FMX DATA

GIVEN: $\phi = .5$ $s = .733$

		β = .05	β = .10	β = .20
α	.01	34	28	22
	.05	24	19	14
	.10	19	15	10
	.20	14	10	7

improve the variance in future evaluation data and thereby reduce the required number of evaluators.

Interpretations of rater sample size requirements is straightforward. It is required that the sample be of sufficient size to insure outcomes that will not be more than .5 distance from what would result if all possible raters were surveyed. For a sample size of 7, the test manager is willing to accept a 20 percent chance of finding that the software is below standard when it is not (Type I alpha error) and a 20 percent chance of finding that the software meets or exceeds the specified standard when it does not (Type II beta error). Although Type I errors (alpha) typically get more attention in the general application of statistics, it is Type II errors (beta) that have the greatest impact in the performance test setting. If software is being tested to determine whether it meets a criteria, the test manager wants as little risk as possible of assuming the criteria was met when, in fact, it was not.

5. Survey Results

As previously mentioned in section B.4, BDM conducted a survey of software professionals to help assess the structure of the AFTEC software maintainability hierarchy, and to specifically determine whether the professionals could agree on the importance of the maintainability test factors. A copy of the survey is contained in Appendix B.

a. Survey Format

The survey was sent to 200 software professionals in industry and academic institutions who had indicated a particular interest and expertise in some facet of software quality measurement. This interest and expertise was determined on the basis of personal BDM acquaintance with the professionals (approximately 40 to 50) and on the basis of recent publications in the literature. The literature reviewed included (among other material):

(1) Raymond T. Yeh. Current Trends in Programming Methodology, Vol. I, Software Specification and Design, Prentice-Hall, 1977.

(2) Robert C. Tausworthe. Standardized Development of Computer Software, Prentice-Hall, 1977.

(3) Computer Software Engineering Symposium Proceedings. Polytechnic Press, New York, 1976.

(4) Defense Documentation Center (DOD/NTIS) Subject Search, December 1977.

(5) Winter Simulation Conference Proceedings, National Bureau of Standards, Maryland, 1977.

The primary objectives of the survey were:

(1) Determine whether a well-qualified group of software professionals could have any reasonable agreement as to what importance (weight) a given test factor should have relative to overall software maintainability.

(2) Solicit comments on the AFTEC software maintainability test factors and their hierarchial relationship to maintainability of software.

Each participant was asked to return a postcard (see Appendix B) with the appropriate scale response (1-lowest importance, 10-highest importance) next to each test factor in the documentation and design categories, respectively (see figure 2). Comments were also solicited, although the space was limited. The responses were meant to be anonymous and the design of the survey was so that each respondent might not have to spend any lengthy period of time completing the responses. There was some concern that too much detail might discourage responses, yet enough detail had to be included so that the overall terminology had a valid chance of being understood.

b. Survey Statistics

Table 3 summarizes the survey response statistics. Most of the responses had been received within 6 weeks of the mailing date. The percentage of responses is considered to be satisfactory in view of the nature of the survey (anonymous, no follow-up). The most suprising statistic was the number of respondents who made at least one comment, as well as a seemingly genuine level of interest (though doubtful of solution) of the respondents. Comments were generally scribbled all

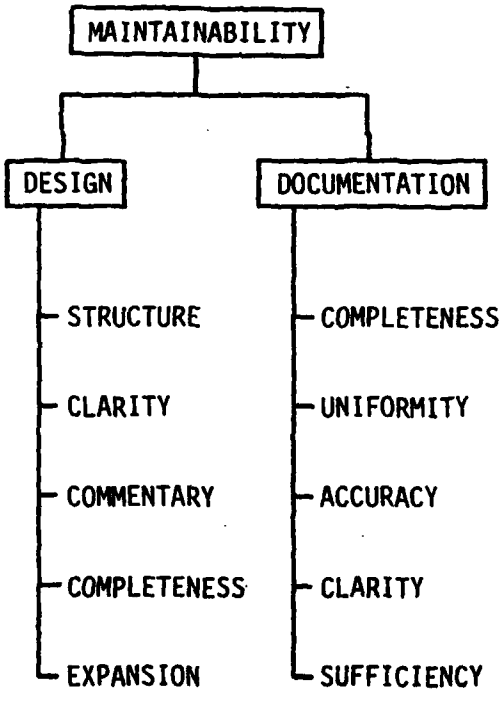


Figure 2. AFTEC Software Maintainability Hierarchy

TABLE 3. SURVEY RESPONSE SUMMARY

COUNT	DESCRIPTION
200	TOTAL NUMBER OF QUESTIONNAIRES MAILED
10	NUMBER OF QUESTIONNAIRES NOT DELIVERED
74	TOTAL NUMBER OF RETURNED RESPONSES
8	NUMBER OF INVALID RETURNED RESPONSES 2 - RANKED RESPONSES 2 - ENTERED INVALID NUMERIC DATA(0) 1 - ALL 10 RESPONSES 3 - MISSING DATA
66	NUMBER OF VALID RETURNED RESPONSES 36.3% (66/182 *100%)
48	NUMBER OF RETURNED RESPONSES WHICH INCLUDED COMMENTS
10	NUMBER OF INDIVIDUAL REQUESTS FOR COPY OF SURVEY RESULTS

over the return postcard and several responses came with a lengthy letter pointing out everything from inconsistencies in the survey itself, to more desirable test factors, to questions concerning the complete details of the evaluation process, and to relevant literature for other software maintainability related methodology. Although the statistic is not mentioned in the table, there were many more than the ten who requested copies of the survey result who included their name. Anonymity, at least for those who responded, did not seem to be an issue.

Figure 3 is an annotated listing of all the postcard numerical responses which were received (there were three additional letter responses with missing postcards). Table 4 is a sampling of some of the comments from the respondents, paired to the numerical response data through the specified response card number.

c. Survey Analysis

The analysis of the survey numerical data was done using the BMDP statistical package available through the AFWL computer center. The original plans were to use many of the statistical packages depending on the basic univariate statistics such as the standard deviation. Since the standard deviation was so large in general, the only analysis packages run were the factor analysis (QRMAX and VMAX). The reader is referred to reference 8 for a more detailed discussion of the BMDP statistical analysis program in general and the factor analysis method and interpretation in particular.

The summary of univariate statistics generated from the input of the data array shown in figure 3, less the five invalid response cards is shown in figure 4. The significant statistics are the large standard deviations and the range of responses from small (1 or 2 in most cases) to large (10) across all the test factors. This clearly indicates the lack of agreement among the respondents as to which test factors were more important and which were not so important in relation to maintainability. With such a wide variation of responses it was clear that no universal set of weights for these test factors could exist. Also, there was no use in exercising other more stringent statistical analysis such as regression/correlation.

RESPONSE NUMBER N	DOCUMENTATION TEST FACTORS CO	UN	AC	CL	SU	DESIGN TEST FACTORS ST	CL	CM	CO	EX	COMMENT CODE 0,1,2,3
1	5	5	8	8	6	10	9	7	8	7	2
2	8	5	10	8	7	8	10	5	7	10	2
3	9	6	10	9	10	8	10	8	10	6	1
4	5	4	6	6	5	8	10	6	7	8	0
5	8	7	10	8	8	10	10	5	8	7	1
6	10	10	10	10	9	10	10	8	2	8	1
7	10	5	10	7	10	8	5	5	9	10	1
8	8	6	10	10	7	10	10	10	10	5	3
9	10	3	8	6	8	5	9	8	10	10	0
10	10	8	8	8	9	9	9	8	10	7	0
11	4	2	9	10	6	8	10	4	8	6	0
13	10	1	5	10	8	2	8	8	10	2	0
14	5	5	10	10	5	10	10	5	5	8	0
15	9	1	10	7	10	10	7	4	5	10	0
16	7	3	10	5	10	5	10	1	5	6	1
17	0	0	0	0	0	0	0	0	0	0	3
18	7	6	9	7	7	8	9	8	7	7	0
19	10	5	10	8	10	10	8	5	3	3	0
20	7	5	7	6	7	9	8	9	10	7	0
21	8	6	10	10	8	10	10	8	5	5	0
22	0	0	0	0	0	10	5	10	10	10	3
23	10	5	10	3	8	10	6	4	10	8	0
24	10	6	8	7	9	10	9	8	6	5	0
25	4	3	2	3	2	10	10	7	9	8	1
26	8	6	10	10	8	8	10	8	2	8	0
27	9	5	10	10	10	7	10	8	5	9	1
28	8	8	10	10	8	8	10	8	8	8	1
29	10	10	10	10	10	10	10	10	10	8	1
30	5	9	10	10	6	9	10	7	10	5	1
31	6	2	5	10	7	9	10	6	8	7	0
32	7	6	10	10	6	10	9	5	10	2	1
33	3	2	8	5	5	5	7	7	10	5	0
34	1	4	2	3	0	0	2	0	1	3	1
35	5	5	7	8	7	7	10	6	8	7	1
36	6	4	10	7	8	8	10	6	9	4	0
37	7	4	8	10	8	7	10	8	6	7	0
38	5	2	10	5	7	5	9	9	2	7	1
39	5	5	10	10	5	10	10	5	5	5	0
40	6	3	6	7	4	10	5	7	5	7	1
41	9	9	10	9	9	10	9	8	9	10	2
42	10	7	8	10	8	10	8	7	10	9	1
43	9	10	10	9	9	10	9	9	9	10	1
44	10	7	10	10	7	7	10	7	9	7	0
45	7	6	7	10	9	10	10	9	9	7	1
46	8	6	9	8	7	8	8	7	6	9	0
47	9	8	9	10	8	9	7	7	8	6	2
48	8	7	8	10	7	10	9	5	9	9	1
49	7	10	10	6	7	10	10	6	8	9	0
50	6	7	7	10	6	10	10	10	8	8	1
51	10	3	9	6	9	10	7	6	9	10	2
52	7	5	3	6	2	10	6	3	7	5	1
53	9	8	10	8	10	6	7	8	3	5	1
54	1	2	10	7	7	10	7	7	6	5	1
55	1	1	9	9	9	9	5	7	9	5	1
56	6	6	5	7	6	9	9	7	6	6	0
57	6	5	10	9	8	10	10	6	8	9	2
58	6	6	6	10	6	10	6	6	8	2	0
59	8	8	10	9	9	9	10	7	9	8	1
60	8	1	7	7	5	8	7	7	3	5	1
61	10	8	10	8	8	8	8	8	10	5	1
62	10	5	10	5	5	7	7	7	7	7	3
63	10	10	10	10	10	10	10	10	10	10	3
64	9	7	10	8	8	5	7	10	8	5	0
65	6	4	8	5	8	7	8	2	4	7	0
66	6	5	7	9	8	7	10	9	8	5	0
67	5	1	10	10	5	10	10	5	1	8	1
68	7	5	9	8	7	8	8	10	7	9	2
69	2	1	6	7	10	9	8	4	5	3	3
70	9	8	10	10	7	8	9	8	7	9	3
71	5	9	8	8	6	9	10	8	5	9	1

NOTES: 1. COMMENT CODES
0 = NO COMMENT
1 = NORMAL COMMENT
2 = PARTICULARLY POSITIVE COMMENT
3 = PARTICULARLY NEGATIVE COMMENT
2. RESPONSE NUMBERS 17,22,30,42,63 HAVE INVALID RESPONSES AND ARE TO BE REMOVED FROM THE DATA ANALYSIS.
3. CO = COMPLETENESS, UN = UNIFORMITY, AC = ACCURACY, CL = CLARITY, SU = SUFFICIENCY, ST = STRUCTURE, CM = COMMENTARY, EX = EXPANSION

Figure 3. Survey Response Data

TABLE 4. SAMPLE SURVEY COMMENTS

RESPONSE CARD NUMBER	COMMENTS (PARAPHRASED)
16	Expansion depends entirely on the proposed use of the system.
22	Documentation - motherhood statements. Two most important factors are simplicity of design and simplicity and unity of "concept:" e.g., UNIX operating system.
25	Documentation should be built into the source code. Clarity is most important.
27	Documentation should be built into the source code. Clarity is most important.
30	Design is never complete, it is "improved" until it can neither be maintained nor used.
32	A means to force conformity and adherence to requirements is needed.
34	Structure and modularity make clarity and expansion possible.
35	There is a misplaced emphasis on documentation. If code is well-commented it will suffice for detailed documentation. What is needed is overall view of system and how it is structured (decomposition, interfaces). Redundancy: Structure, clarity, commentary.
Letter 1	DESIGN is more important than DOCUMENTATION. Top four of DESIGN (as were ranked) are much more important than commentary and DOCUMENTATION.
38	Structure is the use of an appropriate modular decomposition. Your "structure" seems less important.
41	Structure is by far the most important.
42	Redundant: Completeness, sufficiency; structure, clarity, commentary.
43	Modularity is omitted and this is the most important for both design and documentation.

TABLE 4. SAMPLE SURVEY COMMENTS (Continued)

RESPONSE CARD NUMBER	COMMENTS (PARAPHRASED)
45	Redundant: Completeness, sufficiency; structure clarity, commentary.
50	Unlikely any factor will be rated 6 so perhaps scale (1 to 10) is inaccurate. Better to use out of a budget of 100%, how important are factors (rank).
Letter 2	Design and documentation can't be assessed independently. System design is embodied only in documentation (code and associated materials). Six distinguished factors (by rank): structure, clarity, expansion, sufficiency, completeness, uniformity. Sufficiency might be number 2.
52	Code should be prime vehicle for documentation. Written documentation tends to be inaccurate. Several categories overlap.
53	DESIGN is defined (by virtue of factors) as module design. What about program design?
54	No "checklist" will be effective. More "structured" programming is necessary.
55	Contracts often require too much documentation and formatting.
57	Survey validity problem. Accuracy most important for propagation reasons. Structure and clarity are most important while others are "derived". Design completeness - "Defensive Programming" - Kernighan and Plauger.
58	Are metric measures possible?
59	Survey validity problem. Documentation order of factors is not the same on postcard and in definitions.
60	Data structure design and documentation most important but not mentioned.
61	Maintainability is dependent upon organization, management, priority.

TABLE 4. SAMPLE SURVEY COMMENTS (Concluded)

RESPONSE CARD NUMBER	COMMENTS (PARAPHRASED)
Letter 3	Questions vague. Can't complete request.
67	Structure is not as in definition. Structure is primarily modularity.
71	Clarity is to system analyst as commentary is to programmer.

UNIVARIATE SUMMARY STATISTICS

VARIABLE	MEAN	STANDARD DEVIATION	COEFFICIENT OF VARIATION	SMALLEST VALUE	SMALLEST STANDARD SCORE	FIRST CASE FOR SMALLEST	LARGEST VALUE	LARGEST STANDARD SCORE	FIRST CASE FOR LARGEST
1 DOCCMPLT	7.12121	2.31061	.324469	1.0000	-2.65	50	10.0000	1.25	6
2 UNFMTY	5.22727	2.44820	.468352	1.0000	-1.73	13	10.0000	1.95	6
3 ACCRCY	8.59091	1.92935	.224580	2.0000	-3.42	23	10.0000	.73	2
4 DOCCLRTY	8.09091	1.87064	.231203	3.0000	-2.72	21	10.0000	1.02	6
5 SFFCNCY	7.40909	1.82274	.246014	2.0000	-2.97	23	10.0000	1.42	3
6 STRCTR	8.53030	1.72064	.201709	2.0000	-3.80	13	10.0000	.85	1
7 DESCLRTY	8.72727	1.51450	.173536	4.0000	-3.12	54	10.0000	.84	2
8 CMNTRY	6.83333	1.92620	.281883	1.0000	-3.03	15	10.0000	1.64	8
9 DESCMPLT	6.92424	2.43886	.352220	1.0000	-2.43	62	10.0000	1.26	3
0 EXPNSN	6.76788	2.10870	.310657	2.0000	-2.27	13	10.0000	1.52	2

Figure 4. Survey Univariate Statistics

The command input file to the BMDP factor analysis package using the QRMAX method of rotation is shown in figure 5. This file along with the survey response numerical data input resulted in the factor analysis statistics shown in figure 6. Note that most (all but DOCCLRTY) of the documentation test factors do group as factor 1, but that only 21% of the variance (2.117 divided by the # variables times 100%) is accounted for by factor 1 and that approximately 65% of the variance is accounted for by the indicated four factors.

The command input file to the BMDP factor analysis package using the VMAX method of rotation (a more stringent analysis for factors) is shown in figure 7. This file along with the survey response numerical data input resulted in the factor analysis statistics shown in figure 8. In this analysis only one factor accounting for 19% of the variance could be determined. Note that the top five factor loadings are for the Documentation Test Factors. Hence, there is some statistical validation that the specified test factors under documentation do in fact (according to the 66 response cases) give a measure of the maintainability of software from a documentaton evaluation. However, there clearly was not a similar situation for the DESIGN side of the hierarchy.

In order to understand even better why the standard deviation was so high, why the documentation test factors seemed to group together, and why the design test factors did not group, one can look at the response comments (see table 4). Notice that there is a reluctance to accept DESIGN, as defined, being separate from DOCUMENTATION (#30, 45, Letter 2, 53). Also, note the number of comments on the overlap of definitions of the DESIGN test factors (#35, 42, 45, 52, 57), and the comments which simply disagreed with some aspect (definition, exclusion, etc.) of the DESIGN test factors (#46, 22, 34, Letter 1, 38, 43, Letter 2, 60, 67, 71). This set of comments then might indicate that the respondents did understand the Documentation Test Factor definitions while either misunderstanding or disagreeing with the Design Test Factor definitions. Hence, it is reasonable to understand why the Design Test Factors did not group together.

```
BMDP4M - FACTOR ANALYSIS - DOUBLE PRECISION VERSION          PROGRAM REVISED JULY 7, 1975
HEALTH SCIENCES COMPUTING FACILITY                           MANUAL DATE -- 1975
UNIVERSITY OF CALIFORNIA, LOS ANGELES


 PROGRAM CONTROL INFORMATION

      PROBL    TITLE='SURVEY FACTOR ANALYSIS'./
      INPUT    VARIABLE=10.
               FORMAT='(5X,10(3X,F2.0))'.
               CASE=65.
               UNIT=8./
      VARIABLE NAME=DOCCMPLT,UNFMTY,ACCRCY,DOCCLRTY,SFFCNCY,STRCTR,DESCLRTY,
               CMNTRY,DESCMPLT,EXPNSN./
      PRINT    CASE=66.
               STANDARD.
               COVARIANCE./
      PLOT     INITIAL=5.
               FINAL=5.
               FSCORE=5./
      FACTOR   NUMBER=10.
               METHOD=PCA.
               COMMUNALITY=UNALT./
      ROTATE   METHOD=QRMAX./
      END/
```

Figure 5. Survey QPMAX Input Command File

SORTED ROTATED FACTOR LOADINGS (PATTERN)

		FACTOR 1	FACTOR 2	FACTOR 3	FACTOR 4
SFFCNCY	5	.806	0.000	0.000	0.000
ACCRCY	3	.746	0.000	0.000	0.000
DOCCMPLT	1	.708	0.000	.332	0.000
DOCCLRTY	4	0.000	.849	0.000	0.000
DESCLRTY	7	0.000	.591	0.000	.401
DESCMPLT	9	0.000	0.000	.776	0.000
CMNTRY	8	0.000	.376	.635	0.000
UNFMTY	2	.420	.347	.512	.379
EXPNSN	10	.342	0.000	0.000	.746
STRCTR	6	0.000	0.000	0.000	.649
	VP	2.117	1.535	1.451	1.378

Figure 6. Professional Survey QPMAX Factor Analysis

```
BMDP4M - FACTOR ANALYSIS - DOUBLE PRECISION VERSION          PROGRAM REVISED JULY 1, 1975
HEALTH SCIENCES COMPUTING FACILITY                           MANUAL DATE  MAY 1975
UNIVERSITY OF CALIFORNIA, LOS ANGELES


 PROGRAM CONTROL INFORMATION

 /PROBL    TITLE='SURVEY FACTOR ANALYSIS'./
 /INPUT    VARIABLE=10.
           FORMAT='(5X,10(3X,F2.0))'.
           CASE=66.
           UNIT=8./
 /VARIABLE NAME=DOCCMPLT,UNFMTY,ACCRCY,DOCCLRTY,SFFCNCY,STRCTR,DESCLRTY,
           CMNTRY,DESCMPLT,EXPNSN./
 /PRINT    CASE=66.
           STANDARD.
           COVARIANCE./
 /PLOT     INITIAL=5.
           FINAL=5.
           FSCORE=5./
 /FACTOR   NUMBER=10.
           METHOD=PFA./
 /ROTATE   METHOD=VMAX./
 /END
```

Figure 7. Survey VMAX Imput Command File

SORTED ROTATED FACTOR LOADINGS (PATTERN)

		FACTOR 1
UNFMTY	2	.729
DOCCMPLT	1	.585
ACCRCY	3	.561
DOCCLRTY	4	.416
SFFCNCY	5	.495
STRCTR	6	0.000
DESCLRTY	7	0.000
CMNTRY	8	.378
DESCMPLT	9	0.000
EXPNSN	10	.303
	VP	1.906

Figure 8. Survey VMAX Factor Analysis

These survey results were influential in moving BDM to consider more strongly the modification of the AFTEC software maintainability hierarchy in order to eliminate some of the apparent misunderstandings and to strengthen the hierarchy. This could be done by identifying more independent and better defined test factors for software products (documentation and source listings).

d. Survey Conclusions

The conclusions of the survey are summarized below:

(1) There was an adequate number of respondents.

(2) The interest of the respondents was very good.

(3) The variance of the respondent numerical data was much too large for there to be a universal set of "weights" for the test factors.

(4) There seemed to be a better understanding of the Documentation definitions then of the Design definitions: the Design definitions were overlapping.

(5) The Documentation test factors did tend to group together as evidenced by factor analysis.

(6) Due to disagreement with the hierarchy and lack of understanding of that structure, consideration of a better hierarchy should be considered.

D. GUIDELINES FOR FUTURE EVALUATIONS

1. Sample Sizes and Software Selection Process

The sample size of evaluators and program modules is of concern in order that a software evaluation be conducted with optimum use of resources. Neither the amount of data nor the time to analyze all aspects has been adequate to determine that there is or is not an optimal selection scheme.

From the analysis presented in sections B and C, it appears that a set of five evaluators is marginal. However, with careful control of the evaluation process and the availability of automated means for

processing the evaluation results (and hence the increase in analysis capabilties), AFTEC should be able to arrive at an accurate software maintainability evaluation in most cases using the minimum of five evaluators.

The selection of program modules for evaluation from the set of all program modules is a task which is first concerned with defining what is to be considered a module and second with selecting a sample of modules to be evaluated. The following guidelines were used for the specification of modules and the subsequent selection process for the three evaluations (eight programs) in which BDM was a participant. These guidelines also seem to reflect several aspects of Air Force software standardization and the software standardization and the software structure of the type of application programs with which AFTEC is tasked to evaluate.

a. Module Selection Guideline 1 (Define Module)

In defining the level of the module, as much as is possible, the module should be the smallest separately invoked unit of code (the subroutine, procedure, routine, etc.). The usual structure of the software program will then be a collection of components, each performing a major function and composed of several modules.

b. Module Selection Guideline 2 (Stratify Population of Modules)

Assuming there is a set of modules from subsection a above which have been defined as constituting the software program to be evaluated, the next guideline is to stratify the population of modules into natural groups. There are several methods of stratifying the population. Two natural methods which can be used separately or together are to stratify by component (as naturally defined in the program documentation) and/or by core size (order all modules by core size, break into relatively equal sized groups). In the stratification process, the object is to group the population objects by some set of common characteristics, and then sample from each of the stratified groups to obtain a more representative and (hopefully) more statistically valid sample. For AFTEC purposes, the component and core size stratification schemes seem to be adequate.

c. Module Selection Guidelines 3 (Select Sample of Modules)

Assuming that stratification into groups has been done as in 1.b a sample of modules is selected from each stratified group. If no other information is available, then select an equal percentage of modules from each group, at least one (preferably two), with a minimum of 10% selected.

Normally, more information concerning the stratified groups will be available. For the specific evaluations in which BDM participated (at least F-16 and E-4B), each stratified group was the naturally defined program component and within each component there were some other natural groupings such as modules in HQL versus modules in ASSEMBLY or modules grouped by core size (usually two groups, "large" and "small"). Furthermore, one typical component might be utility modules (or math support modules, etc.). This utility component frequently had many modules most of which were small and would have little likelihood of being changed. In this case, fewer modules were selected from this component. In a similar manner, those components having the most significant application programming impact might have one or two more modules selected for evaluation.

Once the complete stratification of the modules by using all available information has been completed (as a necessary supplement to that in 1.b), and the number of modules from each group has been specified, then the modules are randomly selected from each of the groups. For example, one group may have 10 modules with 2 selections while another group has 16 modules with 3 selectons. The modules in each group are numbered 1 to 10 and 1 to 16, respectively. In the first case, 2 random numbers from 1 to 10 are selected (hence selecting the modules). Similarly, the three modules in the second group are selected. This is repeated until all groups have been processed. The resulting set of selected modules is the example of program modules to be evaluated.

As an additional precaution against a particularly glaring oversite, if there are personnel who are in fact already familiar with parts of the software, these personnel can be queried as to the representativeness of the selected sample. This is notto imply that these

personnel should then directly influence the removal of one module, insertion of another, etc. The intent here is to eliminate major sampling errors which may be glaringly apparent to someone more intimately familiar with the software than a brief look at the software hierarchy in the program documentation can afford.

2. Software Maintainability Evaluation Phases

The six phases of the software maintainability evaluation are summarized in figure 9. In this figure, the materials which will be required for each phase by the Software Assessment Team (SAT) evaluation coordinators (e.g., AFTEC personnel) and evaluators (e.g., site maintenance personnel) are specified as well as the time to complete the various aspects of the phase is specified. The time figures are reasonable estimates which are based on the evaluations in which BDM has participated. Note that 1 day is equal to 8 hours.

The six phases as described are fairly self-explanatory. However, there is some key emphasis which should be made. First, there is not allotment of time for travel in any of the time estimated in figure 9. Second, it is envisioned that the review of questionnaires/ guidelines which is specified as part of Phase I will not be necessary since the Evaluator Guideline Handbook is essentially complete and self-explanatory in itself. Misunderstandings should be able to be resolved during Phase III, Calibration Debriefing. Thus, the evaluators would be mailed the evaluation information necessary for Phase II, Calibration Test, and would mail the responses back to the SAT evaluation coordinators for analysis (via Software Maintainability Analysis Program). This would save considerable expense in travel and labor over an extended number of program evaluations.

Figure 10 is an example of the typical program identification information which is the output of the SAT evaluation coordinator effort during Phase I, and input to Phases II, IV, and V. It is emphasized that the only data required by the Software Maintainability Analysis Program

PHASE 1: Pre-Test Software Review

Materials Required: (1) Software Program Documentation
(2) Evaluator Guidelines Handbook

Time Required: (1) Review of Software Program Documentation for the selection of modules and assignation of program identification information (1/2 day)

(2) Review of questionnaires/guidelines with evaluators (1/2 day)

PHASE 2: Conduct Calibration Test

Materials Required: (1) Evaluator Guidelines Handbook
(per evaluator) (2) Evaluator Biodemographic Questionnaire
(3) Software Program Documentation
(4) Calibration-Test-Module Source Listing
(5) AF Form 1530 (comment answer form)
AFTEC Form 92 (evaluation response form)
(6) Software Program Identification Information

Time Required: (1) Review of Evaluator Guidelines Handbook and
(per evaluator) Completion of Evaluator Biodemographic Questionnaire (1/2 day)
(2) Complete Documentation Questionnaire (1/2 day)
(3) Complete Module Source Listing Questionnaire (1/2 day)

PHASE 3: Conduct Calibration Debriefing

Materials Required: (1) Evaluator comment and response answer sheets from Calibration Test
(2) Results from running SMAP on Calibration Test evaluator responses (or manual calculations)

Time Required: (1) Process Calibration Test evaluator responses (1 day)
(2) Review Calibration Test results with evaluators (1/2 day)

Figure 9. Software Maintainability Evaluation Phases

PHASE 4: Complete Software Program Evaluation

Materials Required: (1) Evaluator Guidelines Handbook
(per evaluator) (2) Software Program Documentation
(3) Evaluation Modules Source Listing
(4) AF Forms 1530, AFTEC Form 92
(5) Software Program Identification Information

PHASE 5: Analyze Evaluation Data

Materials Required: (1) Evaluation Data - Evaluator response, comment, biodemographic data
(2) Software Program Identification Information

Time Required: (1) Manually keypunch comment and biodemographic data (1/2 day)
(2) Process AFTEC Form 92 answer sheets through optical scanner (1/2 day)
(3) Analyze SMAP reports (1 day)

PHASE 6: Complete Evaluation Report

Materials Required: (1) Analysis results from PHASE 5.

Time Required: (1) Write Evaluation Report (1 day)

Figure 9. Software Maintainability Evaluation Phases (Concluded)

E-4B

SYSTEM:	NAME = E-4B	ID = 03
SUBSYSTEM:	NAME = OFP	ID = 01
PROGRAM:	NAME = MOCP	ID = 01
EVALUATORS:	NAME = Mosora	ID = 001
	NAME = Robinett	ID = 002
	NAME = Rowe	ID = 003
	NAME = Baur	ID = 004
	NAME = Storla	ID = 005

DOCUMENTATION:

<AS APPROPRIATE>

MODULES:

ID = 01	Dynamic Check Sum Routine (B2DYNCS)
ID = 02	Interrupt Handler Routine (B1BINTHD)
ID = 03	Startup/Restart (X1STARTS)
ID = 04	Exit ESR Routine (X2EXITER)
ID = 05	Journal ESR Routine (X2JOURNAL)
ID = 06	Queue ESR Routine (X2QUEUER)
ID = 07	Abnormal Condition ESR Routine (X2ABCOND)
ID = 08	I/O Interrupt Preprocessor (X1IOPREP)
ID = 09	Keyboard Printer I/O Complete Handler (X2KBDPTR)
ID = 10	Timekeeper (X1TIMEKP)
ID = 11	Scheduler (X1SCHEDU)
ID = 12	System Error Processor (X1EMPHLR)
ID = 13	Interrupt Return Processor (X2IRPROC)
ID = 14	Task Dispatcher (X3DISPAT)
ID = 15	Insert DTQ TQE Subroutine (XCIDQTQE)
ID = 16	Abnormal I/O Complete Subroutine (XCABNORM)
ID = 17	Timer Subroutine (XCTIMERS) *Calibration Test Module
ID = 18	Journal Suspension Task (S1JOURNL)
ID = 19	Online Confidence Task (S1CHECKS)
ID = 20	Test I/O Device Subroutine (S2TIODEV)
ID = 21	Keyboard Printer Test Subroutine (S3KBTEST)
ID = 22	Convert Mnemonic Subroutine (MCONVERT)
ID = 23	Online Command Routine (M2ONLINE)
ID = 24	MSGCP Indicator Control Task (M1MCPIND)
ID = 25	AUTODIN Link Control Task (A1LINKCN)
ID = 26	Data Output Subroutine (A2TOUT)
ID = 27	Transient Termination (A1TTERM)
ID = 28	Partial Read Subroutine (ACTPART)
ID = 29	Message Block Read Subroutine (ACRRDBLK)
ID = 30	Line Post Print (AIRPOST)
ID = 31	Follow-on Print Subroutine (ACFOPRNT)
ID = 32	Page Post Print Task (A1PGEFO)
ID = 33	Log Output Subroutine (ACLOG)

Figure 10. Software Program Identification Information

(other t uator biodemographic, comment, and evaluation response
data) i s name, ID information for the system, subsystem
program. d evaluators.
of the time required for a typical evaluation in
terms of is summarized in table 5. This table does include
estimates . The critical variable inputs are the number of
modules, of evaluators, and the number of SAT evaluation
coordinat otal number of labor-days is estimated to be 78.25.
If the e re dedicated with no interrupti , then this 78.25
labor da spread over 20.25 physically sequential days, or
approxi nth of four 5-day weeks.

TABLE 5. EVALUATION TIMETABLE EXAMPLE

PARAMETERS:

NE = NUMBER OF EVALUATORS = 5
NM = NUMBER OF MODULES = 30
NC = NUMBER OF SAT EVALUATION COORDINATORS = 1
TD = DAYS TDY PER TRIP = 1

PHASE	STEP	SAT COORDINATORS	TDY	EVALUATORS	TOTAL(LABOR DAYS)
1	1	NC*(1/2)	0	0	0.5
	2	NC*(1/2)	NC*(1)	NE*(1/2)	4.0
2	1	0	0	NE*(1/2)	2.5
	2	0	0	NE*(1/2)	2.5
	3	0	0	NE*(1/2)	2.5
3	1	NC*(1)	0	0	3.0+
	2	NC*(1/2)	NC*(1)	NE*(1/2)	4.0
4	1	0	0	NE*NM*(3/8)	56.25++
5	1	NC*(1/2)	0	0	.5
	2	NC*(1/2)	0	0	.5
	3	NC*(1)	0	0	1.0
6	1	NC*(1)	0	0	1.0
ALL	ALL	7.5	2.0	68.75	78.25

+ = TWO LABOR DAYS ARE ADDED HERE FOR OVERHEAD OF MANUALLY KEYPUNCHING DATA AND PROCESSING QUESTIONNAIRE ANSWER SHEETS THROUGH THE OPTICAL SCANNER

++ = A FIGURE OF 3 HOURS PER MODULE IS USED

APPENDIX A
RELIABILITY DATA

TABLE A-1. AVERAGE RELIABILITIES FOR AFTEC FORM Q-246 ON E-3A AND F-16 EVALUATIONS

1. .82	17. .83	33. .34
2. .58	18. .77	34. .76
3. .60	19. .85	35. .71
4. .53	20. .80	36. .81
5. .68	21. .70	37. .85
6. .69	22. .77	38. .82
7. .64	23. .69	39. .71
8. .57	24. .81	40. .83
9. .74	25. .83	41. .68
10. .70	26. .66	42. .85
11. .85	27. .65	43. .73
12. .68	28. .73	44. .71
13. .65	29. .65	45. .84
14. .79	30. .71	46. .71
15. .68	31. .75	47. .72
16. .67	32. .73	48. .73
		49. .79
		50. .76

SUMMARY

	BELOW 50	50-59	60-69	70-79	80-89	90-99
NUMBER	1	3	13	20	13	
PERCENT	.02	.06	.26	.40	.26	0
CUM PERCENT	.02	.08	.34	.74	1.0	1.0

TABLE A-2. RELIABILITIES FOR AFTEC FORM Q-246 ON B-52 CPT EVALUATIONS

1.	.590	17.	.179	33.	*
2.	*	18.	*	34.	*
3.	*	19.	*	35.	*
4.	.250	20.	.769	36.	*
5.	*	21.	.528	37.	*
6.	.678	22.	.640	38.	.323
7.	.799	23.	.950	39.	*
8.	.422	24.	.734	40.	.620
9.	.647	25.	.721	41.	.736
10.	.914	26.	.829	42.	.866
11.	.846	27.	.813	43.	.858
12.	*	28.	.814	44.	*
13.	.253	29.	.201	45.	.943
14.	*	30.	*	46.	.763
15.	.873	31.	*	47.	.773
16.	.737	32.	.702	48.	*
				49.	*
				50.	*

* NOT INTERPRETABLE - RELIABILITIES OF ZERO OR ONE ARE NOT INCLUDED IN THE AVERAGE

SUMMARY

	BELOW 50	50-59	60-69	70-79	80-89	90-100
NUMBER	6	1	5	7	8	3
PERCENT	.2	.03	.17	.23	.27	.10
CUM PERCENT	.2	.23	.40	.63	.90	1.0

TABLE A-3. RELIABILITIES FOR THE REDESIGNED SOFTWARE MAINTAINABILITY SOURCE LISTING QUESTIONNAIRE ON THE F-16 FMX EVALUATION

1.	.875	23.	.867	45.	.889	67.	.001
2.	.833	24.	.456	46.	.556	68.	.667
3.	.545	25.	.133	47.	.934	69.	.762
4.	.926	26.	.816	48.	.930	70.	.533
5.	.111	27.	*	49.	.963	71.	*
6.	*	28.	*	50.	*	72.	*
7.	*	29.	.936	51.	*	73.	.790
8.	.001	30.	.569	52.	.667	74.	.821
9.	.889	31.	.872	53.	*	75.	.951
10.	.557	32.	.242	54.	.821	76.	.936
11.	.778	33.	.971	55.	.252	77.	.305
12.	.745	34.	.588	56.	*	78.	.808
13.	.133	35.	.762	57.	*	79.	.997
14.	.667	36.	*	58.	.784	80.	.985
15.	*	37.	.907	59.	.519	81.	.667
16.	*	38.	*	60.	.833	82.	.987
17.	.115	39.	.847	61.	*	83.	.524
18.	*	40.	.947	62.	.853	84.	.870
19.	.605	41.	.333	63.	.744	85.	.730
20.	.001	42.	*	64.	.841	86.	.819
21.	*	43.	.533	65.	.680	87.	.767
22.	.938	44.	.714	66.	.930	88.	.959
						89.	.814

SUMMARY

	BELOW 50	50-59	60-69	70-79	80-89	90-100
NUMBER	10	9	6	10	17	16
PERCENT	.15	.13	.09	.15	.25	.24
CUM PERCENT	.15	.28	.37	.52	.77	1.0

APPENDIX B
BDM SOFTWARE MAINTAINABILITY
FACTOR IMPORTANCE SURVEY

2600 Yale Blvd., S.E., Albuquerque, NM 87106 · (505) 843-7870

THE BDM CORPORATION

11 January 1978

Dear Colleague:

The BDM Corporation would appreciate your participation in a survey as to the importance of certain assessment factors for software maintainability. BDM is under Government contract to the Air Force Test and Evaluation Center to refine part of a current methodology for evaluating software acquired at the maintenance point in the software's life cycle. This part of the methodology consists of two phases. First, a set of evaluators independently analyze a predetermined subset of a given software package and complete questionnaires designed to determine how well the software has been documented, designed, and coded. Second, the evaluators' answers and biographical data are statistically analyzed to determine which assessment factors pertaining to documentation or to design satisfy a preset measurable threshold. The assessment factors currently being used are described in Attachment 1 under the categories of Design and Documentation.

In order to get a better feel for the usefulness and accuracy of weights applied to these assessment factors we are requesting the opinions of 200 highly qualified professionals. We have briefly described each factor in Attachment 1 and ask that you rate each factor on a scale from 1 to 10 in relative importance. These factors are clearly not the only ones which could have been chosen. And, you may find the explanations of the factors somewhat inadequate. Nontheless, we would appreciate the few minutes of your time that it will take to subjectively assess the importance of each factor and enter your scale value on the enclosed postcard.

Please feel free to add your comments. If you would rather call to submit your comments and/or ratings, or to clarify terminology, BDM's IN WATS line is (800) 545-8304. Thanks for your support.

Yours truly,

THE BDM CORPORATION

Dave Peercy

Dr. David E. Peercy

DEP:gm
Enclosure

SOFTWARE MAINTAINABILITY

A characteristic of software which affects the capability of support personnel to accomplish software maintenance. Maintainability is generally considered a function of design, documentation, and computer support resources.

1. Software Design

Those characteristics of software that enhance the overall modifiability of the program. Test factors for software design include, but are not limited to, structure, clarity, commentary, completeness, and expansion (growth potential) of the software.

a. Structure

The manner in which a software source code has been constructed. It includes conventions used in naming variables, labeling statements, transferring control, nesting of "do loops", placement of commentary, logical flow of code, etc.

b. Clarity

The characteristic of a source code that allows the reader to easily understand the purpose and logic of the code. It is primarily concerned with the quality of the commentary and the simplicity or complexity of the coding structure.

c. Commentary

This consists of the comments placed in the source code listings. Commentary includes the conventions used relative to placement of comments and the identification of comments. Completeness, clarity, usefulness, and quantity of comments are included when evaluating commentary.

d. Completeness

Those characteristics necessary for the source code to stand alone. All routines necessary for the program to operate should be part of the code, with the exception of the routines provided by the standard operating system. Completeness addresses such things as protection from undefined operations, checking of index limits, and error exits.

e. Expansion

How the code has been structured to allow for array expansion, increased data base, and addition of new functions.

2. Software Documentation

The technical data that decribes software programs and their use. Assessment of software documentation includes, but is not limited to, the test factors of clarity, accuracy, uniformity, completeness, and sufficiency.

a. Clarity

The characteristic that allows the reader to easily understand, from the description alone, the purpose and logic of a function (excludes source code listings covered under design). Clarity should not depend on an assumed knowledge of the system. Clarity is a function of definitions and logic flow in presentation as well as the language used.

b. Accuracy

The correlation between the documentation and source code listings.

c. Uniformity

Uniformity of documentation, other than the source code, is the degree to which a convention has been followed in the preperation of the documentation, (e.g., the inclusion and placement of parallel sections from document to document).

d. Completeness

The presence of all required documents and the thoroughness to which the subject is addressed.

e. Sufficiency

The content and quantity of documentation. Documentation must cover all areas of concern and provide the detail needed to assess the impact of proposed modifications to the software and to make modifications, if necessary.

IMPORTANCE OF SOFTWARE MAINTAINABILITY FACTORS

Factors listed below will influence the maintainability of software. Rate the importance of each factor to program maintenance and modification. Use a scale from 1 (very little importance) to 10 (very great importance) on each item. Each factor is to be rated independent of the other factors. Thus several factors could be assigned the same importance number.

DOCUMENTATION		DESIGN	
Completeness	_____	Structure	_____
Uniformity	_____	Clarity	_____
Accuracy	_____	Commentary	_____
Clarity	_____	Completeness	_____
Sufficiency	_____	Expansion	_____

COMMENTS: _____

_____

_____

BDM CORPORATION
2600 Yale Blvd., SE
Albuquerque, NM 87106

ATTN: Dr. D. E. Peercy